공학석사학위논문

텍스트 임베딩을 이용한 퀴즈 자동 생성

Automatic Generation of Multiple-Choice Gap-Fill Quizzes
Using Text Embedding

2017 년 8 월

서울대학교 대학원

컴퓨터공학부

박 정 혁

초 록

퀴즈는 다양한 장소와 분야에서 널리 쓰이고 있다. 퀴즈 쇼나 퀴즈 대회들뿐만 아니라 특정 분야의 능력 평가 및 교육 현장에서도 그 수요가 크다. 하지만 사람이 직접 이러한 퀴즈를 만드는 데에는 많은 시간과 노력이 들어간다. 따라서 자동으로 이러한 퀴즈를 생성해내는 시스템에 대한 요구가 큰 바, 본 논문에서는 자동으로 객관식 빈칸 채우기 퀴즈를 생성하는 방법을 제안하였다.

기존의 객관식 빈칸 채우기 연구는 1) 텍스트의 시맨틱을 경시하고, 2) 도메인에 종속적인 피쳐와 룰을 사용하는 한계점이 있었다. 따라서 본 논문에서는 1)의 문제점을 해결하기 위해 텍스트의 시맨틱을 반영할 수 있는 텍스트 임베딩 모델을 제안하였고, 2)의 문제점을 해결하기 위해 유사도 기반 퀴즈 생성 방법을 제안하였다. 또한 미리 정의된 키워드 리스트가 있는 경우에 대하여 최종적인 퀴즈의 질을 향상시키기 위하여 키워드 기반 사후 필터링 방법을 제시하였다.

데이터베이스 책과 생물학 책에 대해 실험한 결과 기존 객관식 빈칸 채우기 퀴즈 연구에 비해 본 논문의 유사도 기반 퀴즈 생성 방법이 10%p~30%p정도 성능이 더 높았으며, 또한 키워드 기반 사후 필터링 방법을 적용하였을 때 10%p~20%p의 성능 향상을 보여 본 논문의 유사도 기반 퀴즈 생성 방법과 키워드 기반 사후 필터링 방법의 성능 향상 효과를 확인할 수 있었다.

주요어: 자동 퀴즈 생성, 단어 임베딩, 텍스트 임베딩, 신경망 기반 임베딩 모델, 의미론적 유사성

학 번: 2015-22898

목 차

표 목차

그림 목차

제 1장 서론

퀴즈(quiz)란 참가자들이 주어진 질문의 정답을 답하는 게임을 통칭하여 이르는 말이다. 미국의 Jeopardy![1], 영국의 Who Wants to be a Millionaire[2], 한국의 1대 100[3]과 같은 퀴즈 쇼나, Word Quizzing Championship (WQC)[4], European Quizzing Championship (EQC)[5] 등과 같은 퀴즈 대회들도 넓은 의미에서의 퀴즈로도 볼 수 있다. 하지만 우리가 흔히 생각하는 퀴즈는 위의 의미보다는 좁은 의미로, 참가자들에게 주어지는 질문이나 문제 그 자체를 말한다. 앞으로 언급하는 퀴즈의 의미는 후자의 의미로 한정하도록 하겠다.

퀴즈는 다양한 분야와 장소에서 널리 쓰이고 있다. 위에서 언급한 퀴즈 쇼나 대회들에서는 물론이고, 응시자들의 특정 분야에 대한 능력을 평가하는 용도로서 널리 사용된다. 영어 능력을 평가하는 TOEIC[6], TOEFL[7], TEPS[8] 시험, 한국사에 대한 지식을 평가하는 한국사능력검정시험[9], 각종 자격증의 필기시험, 취업을 위한 적성 검사, 대입을 위한 대학

1 https://www.jeopardy.com/

2 http://millionairetv.dadt.com/

3 http://www.kbs.co.kr/2tv/enter/1vs100/

4 http://www.worldquizzingchampionships.com/

5 http://www.europeanquizzingchampionships.com/

6 https://www.ets.org/toeic

7 https://www.ets.org/toefl

8 http://www.teps.or.kr/

9 http://www.historyexam.go.kr

수학능력시험[10] 등 다양한 분야에서의 능력 평가 시험들이 존재한다. 또한 교육 현장에서도 퀴즈는 매우 중요한 역할을 하는데, 중간 그리고 기말고사, 그리고 수업 시간마다 학생들에게 주어지는 깜짝 퀴즈(pop quiz)는 학생들이 수업의 내용을 얼마나 잘 이해하고 있는지 평가하는 가장 쉽고 좋은 방법이다. 특히 최근 일부 대학에서 시행되고 있는 플립드 러닝(flipped learning)[11] 수업에서는 퀴즈가 매우 중요한 역할을 하는데, 이 수업 환경에서는 학생들이 수업시간 전 미리 온라인 강의 동영상을 보고 오프라인 강의에 와서는 심도 있는 토론을 진행하기 때문에, 토론을 원활하게 진행하기 위해서는 매 수업시간 토론 전 퀴즈를 통해 학생들이 온라인 강의 동영상의 내용을 잘 이해했는지 평가할 필요가 있다. 필자도 플립드 러닝으로 진행되었던 2015년과 2016년 데이터베이스 수업의 조교를 하면서 이러한 퀴즈의 중요성에 대해 몸소 느낀 바 있다.

앞서 말한 대로 퀴즈의 수요는 매우 크지만, 이를 만드는 데에는 많은 시간과 노력이 들어간다. 우선 어떤 주제나 개념을 퀴즈로 출제할 것인지 정해야 하고, 어떤 질문을 할 것인지, 또 객관식 퀴즈의 경우 답이 아닌 보기들을 어떻게 정할 것인지 등을 고려해야 하기 때문이다. 그렇기 때문에 많은 자격증 시험들이나 일부 영어 시험들은 문제은행(question

10 http://www.suneung.re.kr/main.do?s=suneung

11 역진행 수업(逆進行 修業, flipped learning) 또는 플립드러닝, 플립러닝, 역전(逆轉)학습, 거꾸로 교실은 혼합형 학습의 한 형태로 정보기술을 활용하여 수업에서 학습을 극대화할 수 있도록 강의보다는 학생과의 상호작용에 수업시간을 더 할애할 수 있는 교수학습 방식을 말한다. 흔히 적용되는 방식으로는 교사가 준비한 수업 영상과 자료를 학생이 수업시간 전에 미리 보고 학습하는 형태가 있다. 그 후 교실 수업시간에 교사는 교과내용을 중심으로 가르치기보다 학생들과 상호작용하거나 심화된 학습활동을 하는 데 더 많은 시간을 할애할 수 있다.

bank) 방식을 채택하여 매번 퀴즈를 만들어내는 수고를 덜고 있지만, 그 문제은행을 만드는데도 많은 노력이 들어가기 때문에 이는 근본적인 해결책이라고 할 수 없다. 따라서 사람의 수고가 들어가지 않는, 즉 기계에 의하여 자동적으로 대량의 퀴즈를 생성하는 시스템의 필요성이 대두되고 있다. 특히 빈칸 채우기 퀴즈의 경우 각종 영어 시험에서의 어휘나 문법 테스트, 각종 퀴즈 쇼, 수업 시작 전 간단히 보는 퀴즈, 각종 이러닝(e-learning) 플랫폼 등에서 다양한 용도로 사용되고 있고 그 수요가 높아 빈칸 채우기 퀴즈를 자동으로 생성하는 시스템에 대한 요구는 특히 큰 상황이다.

본 논문에서는 다양한 텍스트로부터 자동으로 대량의 객관식 빈칸 채우기 퀴즈를 생성하는 자동 퀴즈 생성(Automatic Quiz Generation) 시스템을 제안한다. 기존 연구에서 이용하지 않았던 텍스트의 시맨틱(semantic)을 이용하기 위하여 문서, 문장, 단어 임베딩(embedding)을 학습하기 위한 새로운 임베딩 모델을 제안한다. 또한 이렇게 생성한 임베딩을 이용하여 퀴즈를 생성하는 유사도 기반 퀴즈 생성 방법을 제시한다. 마지막으로 생성된 퀴즈를 사후 필터링(filtering)하여 최종적인 퀴즈의 질을 획기적으로 향상시키는 키워드 기반 필터링 방법을 제시한다. 실험 결과, 기존의 자동 퀴즈 생성 연구에서의 객관식 빈칸 채우기 퀴즈보다 본 연구의 방법에 의해 생성된 퀴즈의 품질이 더 뛰어남을 확인할 수 있었다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 자동 퀴즈 생성 연구와 본 논문에서 핵심적인 역할을 하는 텍스트 임베딩 모델에 대해 이야기하고, 3장에서는 텍스트 임베딩을 이용한 객관식 빈칸 채우기 퀴즈 자

동 생성 방법을 제안한다. 4장에서는 제안한 방법을 이용하여 생성된 퀴즈들의 성능을 평가하고, 5장에서는 본 논문의 결론 및 향후 연구를 제안한다.

제 2장 관련 연구

2.1 자동 퀴즈 생성 연구

자동 퀴즈 생성 연구는 자연 언어 처리(Natural Language Processing, NLP) 분야의 한 주제로서 크게 두 갈래로 진행되어 왔다. 첫 번째는 질문형 퀴즈 생성(Wh-Quiz Generation) 연구로서, 주어진 문장을 질문형으로 바꾸는 데에 초점을 맞춘 연구들이 있었다. 두 번째는 빈칸 채우기 퀴즈 생성(Gap-Fill Quiz Generation) 연구로서, 주어진 문장에서 중요한 갭(gap)을 찾아 이 단어를 빈칸으로 치환함으로써 퀴즈를 만들어내는 연구들이 있었다. 앞으로 1절과 2절을 통해 이들 두 갈래의 대표적인 연구들을 간단히 소개하고자 한다.

2.1.1 질문형 퀴즈 생성 연구

[1]은 거의 최초의 자동 퀴즈 생성 연구로 꼽히는데, wh로 시작하는 질문과 그에 대한 보기를 생성하여 다음과 같은 퀴즈를 만들어냈다.

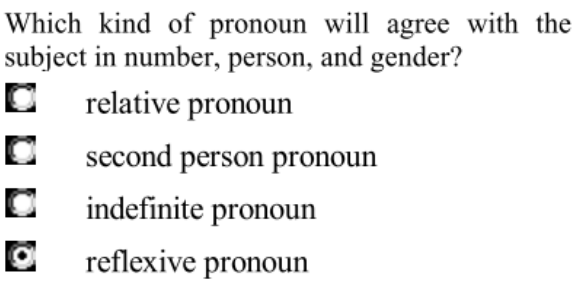

그림 1 [1]에 의해 생성된 퀴즈의 예

이 연구는 먼저 얕은 파싱(shallow parsing)을 이용하여 주어진 문장에서 명사와 명사구들을 추출, 이들을 핵심 단어로 삼는다. 그리고 객관식 보기를 만들기 위하여 추출한 핵심 단어와 의미가 비슷한 단어를 WordNet[2]에서 찾아 보기로서 선택한다. 이 때, 명사구의 경우 코퍼스에서 같은 "head"를 가지고 있는 어구들을 보기로서 선택한다. 마지막으로 주어진 문장과 선택된 핵심 단어를 이용하여 변환 규칙(transformation rule)을 사용해 퀴즈를 생성하였다.

[3]은 가능한 많은 퀴즈를 생성한 후, 랭킹 방법을 이용하여 수용 가능한 범위 내의 퀴즈 만을 선택한다. 이 연구는 먼저 주어진 문장을 파스 트리(parse tree) 형식으로 변환한 후, 미리 정의된 변환 규칙을 이용하여 이들 문장을 단순화, 선언적 문장(declarative sentence)을 생성한다. 그 후 질문 변환(question transducer) 모듈이 선언적 문장으로부터 가능한 모든 정답 어구를 대상으로 변환 규칙을 이용하여 질문을 생성한다. 마지막으로 이렇게 생성된 퀴즈들을 따로 정의한 피쳐(feature)를 이용하여 순위를 매기고, 수용 가능한 범위의 퀴즈들만 받아들인다. 정의한 피쳐들은 모두 좋은 퀴즈가 될 수 없는 요소들로, *ungrammatical*(문법 오류), *does not make sense*(문법은 맞지만 해독할 수 없음), *vague*(무엇을 물어보는지 애매모호함), *obvious answer*(정답이 명백함), *missing answer*(문서에 정답이 없음), *wrong WH word*(WH word가 잘못됨), *formatting*(포맷 에러) 등이 있다.

[4]는 이러닝 플랫폼인 TED Talks[12]에 적용 가능한 TEDQuiz라는 시스

12 https://www.ted.com/

템을 개발하였다. 이 연구는 먼저 크롤러(crawler)를 통해 TED Talks 동영상의 스크립트를 크롤링(crawling)한 후 그래프 기반 자동 요약(Automatic Summarization) 알고리즘 중 하나인 LexRank[5]를 사용하여 스크립트에서 중요한 문장들을 추출하였다. 이렇게 추출된 중요한 문장들은 다른 문장들과 높은 상관관계가 있고, 따라서 해당 스크립트의 주요 주제라고 할 수 있다. 그 후 [3]의 방법을 이용하여 퀴즈를 생성한다. 마지막으로 객관식 퀴즈를 만들기 위해 핵심 단어에 상응하는 보기를 만드는데, 1) 단어의 supersense 2) 품사 태그(Part-of-speech tag) 3) n-gram 정보를 이용한다.

질문형 퀴즈 연구들은 위와 같이 다양한 NLP 기법들과 직접 정의한 룰들을 사용하여 퀴즈를 생성한다. 그렇기 때문에 위의 연구들은 퀴즈 생성 프로세스가 각자 달라 연구의 연속성이 떨어진다는 단점이 있다. 또한 직접 정의한 룰들이 많기 때문에 다른 도메인(domain)에서, 즉 입력 텍스트의 형태나 주제가 달라질 경우에는 잘 동작하지 않을 수 있다는 단점이 있다. 그리고 문법적으로 옳은 문장을 만드는데 집중하기 때문에 텍스트의 시맨틱은 경시하는 경향이 있다.

2.1.2 빈칸 채우기 퀴즈 생성 연구

빈칸 채우기 퀴즈 생성 연구는 질문형 퀴즈 생성 연구와 같이 문장을 질문형으로 변형시키는데 관심을 두지 않고, 문서에서 퀴즈가 될 만한 중요한 문장과 그 문장에서의 중요한 단어를 찾는 데에 중점을 두는 연구를 말한다. 이 연구들은 질문형 퀴즈 연구와는 다르게 공통적으로 다음 세 단계를 따른다.

1. Sentence Selection(문장 선택)

문장 선택 단계에서는 입력으로 들어온 문서에서 퀴즈에 쓸 만한 중요한 문장들을 선택한다.

2. Gap Selection(갭 선택)

갭 선택 단계에서는 문장 선택 단계에서 선택된 중요한 문장에서 퀴즈의 정답이 될 만한 중요한 키워드(keyword)를 선택한다.

3. Distractor Selection(오답지 선택)

오답지 선택 단계에서는 갭 선택 단계에서 선택된 중요한 키워드와 의미적으로 비슷하지만 키워드와 동의어 및 유의어 관계가 아닌 단어들을 오답지, 즉 정답 이외의 보기로 선택한다.

지금부터 설명할 연구들은 객관식 퀴즈의 경우에는 1~3단계로, 객관식 퀴즈가 아닌 경우에는 1~2단계로 이루어져 있다.

[6]은 비 영어권 학생들을 위한 영어 어휘 퀴즈를 생성하는 연구를 진행하였다. 먼저 입력 문장에서 동사를 갭으로 선택한 후, 외부의 유의어 사전(thesaurus)을 이용하여 갭과 비슷한 의미를 가진 단어들을 오답지 후보들로 선택한다. 이렇게 선택된 오답지 후보들을 갭 자리에 바꿔 넣고, 그 단어의 n-gram을 웹에 검색하여 검색 결과가 없는 오답지들을 최종적으로 선택한다. 이 연구는 간단한 방법을 사용하고 있지만 최초의 빈칸 채우기 퀴즈 생성 연구라는 점에 의의가 있다.

[7]은 생물학 교과서로부터 객관식 빈칸 채우기 퀴즈를 생성하는 연구

를 진행하였다. 이 연구에서는 각 단계에서 피쳐 기반 스코어링(scoring) 방법을 사용한다. 다시 말하면, 중요한 문장, 갭, 오답지가 무엇인지를 설명하는 피쳐를 정의하고, 이들에 점수를 매겨 합친 것을 최종적의 점수로 하여 이 점수가 높은 문장, 갭, 오답지를 선택하는 것이다. 표 1은 이 연구에서 사용된 피쳐들의 목록을 나열한 것이다.

표 1 [7]이 사용한 피쳐의 목록

	심볼	설명
문장	$f(s_i)$	Is s_i the first sentence of the document?
	$sim(s_i)$	No. of tokens common in s_i and $title/length(s_i)$
	$abb(s_i)$	Does s_i contain any abbreviation?
	$super(s_i)$	Does s_i contain a word in its superlative degree?
	$pos(s_i)$	s_i's position in the document $(= i)$
	$discon(s_i)$	Is s_i beginning with a discourse connective?
	$l(s_i)$	Number of words in s_i
	$nouns(s_i)$	No. of nouns in $s_i/length(s_i)$
	$pronouns(s_i)$	No, of pronouns in $s_i/length(s_i)$
갭	$term(k_p)$	Number of occurrences of the k_p in the document
	$title(k_p)$	Does title contain k_p?
	$height(k_p)$	Height of the k_p in the syntactic tree of the sentence
오답지	$context(d_p, k_s)$	Measure of contextual similarity of d_p and the k_s in which they are present
	$sim(d_p, k_s)$	Dice coefficient score between GFS and the sentence containing the d_p
	$diff(d_p, k_s)$	Difference in term frequencies of d_p and k_s in the chapter

이 연구는 영어 학습 도메인이 아닌 일반 도메인을 대상으로 한 첫 객관식 빈칸 채우기 퀴즈 연구라는 점에 의의가 있다.

[8]는 위키피디아(Wikipedia) 문서로부터 빈칸 채우기 퀴즈를 생성하는 연구를 진행하였다. 이 연구는 자동 퀴즈 생성 연구 분야에서 최초로 기계학습 알고리즘(machine learning algorithm) 중 하나인 분류(classification) 방법을 사용했다는 것에 의의를 둘 수 있다. 우선 자동 요약 알고리즘의 하나인 SumBasic[9] 알고리즘을 이용하여 중요한 문장을 선택한 후, 모든 명사구와 형용사구를 갭 후보로 하여 가능한 모든 퀴즈를 대량으로 생성한다. 이렇게 생성된 퀴즈에 대해 사람이 좋음, 보통, 나쁨으로 태깅(tagging)을 함으로써 분류기(classifier)를 학습할 수 있는 학습 데이터(training data)를 구축한다. 그리고 미리 정해진 피쳐들과 학습 데이터를 사용하여 로지스틱 회귀(logistic regression)[13]를 학습한다. 이렇게 학습된 모델은 새로운 퀴즈를 질이 좋은 퀴즈와 나쁜 퀴즈로 분류할 수 있게 된다.

[10]는 교과서로부터 객관식 빈칸 채우기 퀴즈를 생성하는 연구를 진행하였다. 문장 선택과 갭 선택 단계는 [8]의 방법에 TF-IDF, Word2Vec[11], WordNet[12], 토픽 분포 등의 피쳐를 추가하는 방식으로 진행되었다. 그리고 오답지 선택 단계는 Word2Vec, WordNet, dice coefficient[14],

[13] 로지스틱 회귀(logistic regression)는 확률 모델로서 독립변수의 선형 결합을 이용하여 사건의 발생 가능성을 예측하는데 사용되는 통계 기법이다. 선형 회귀(linear regression) 분석과는 다르게 종속 변수가 범주형(categorical) 데이터를 대상으로 하며 입력 데이터가 주어졌을 때 해당 데이터의 결과가 특정 분류로 나뉘기 때문에 일종의 분류(classification) 기법으로도 볼 수 있다.

[14] $DC(X,Y) = \frac{2|X \cap Y|}{|X|+|Y|}$

그리고 언어 모델(language model)[15]을 이용하여 갭과의 유사도(similarity)를 측정하는 방식으로 진행되었다.

빈칸 채우기 퀴즈 생성 연구는 질문형 퀴즈 생성 연구와는 다르게 연속성 있게 발전해왔다. 도메인은 영어 학습 도메인에서 일반적인 도메인으로, 방법은 나이브(naïve)한 룰 기반 방법에서 기계학습 알고리즘을 이용하기에 이르기까지 발전해왔다. 하지만 여전히 특정 도메인에 종속적인 룰과 피쳐를 사용하고 있고, 텍스트의 시맨틱을 경시하는 경향이 있다.

2.2 신경망 기반 텍스트 임베딩 모델 연구

텍스트 임베딩 모델이란 일정 단위의 텍스트(단어, 문장, 문서)를 벡터(vector)로 표현하여 텍스트의 의미 관계를 수학적으로 나타낼 수 있게 만드는 모델이다. 텍스트 임베딩 모델은 텍스트를 고정 차원의 실수 벡터로 만들기 때문에 텍스트 사이의 유사도(거리) 측정, 여러 텍스트의 평균 계산 등 텍스트 단위에서는 할 수 없었던 작업을 벡터 연산을 통해 쉽게 할 수 있다. 또한 벡터 연산을 통한 추론 작업도 가능한데, 가령 '한국'이라는 단어에서 '서울'이라는 단어의 의미를 빼고, '도쿄'라는 단어의 의미를 넣는다면 사람은 세 단어에서 '국가'와 '수도' 사이의 관계를 알아채고 '일본'이라는 의미가 될 것임을 추론할 수 있는데, 실제 이들 단어의 벡터를 가지고 같은 연산, 즉 $v('한국') - v('서울') + v('도쿄')$ 연산을

[15] 5-gram Kneser Ney Back-off Language Model을 사용하였다.

수행하면 결과 벡터와 가장 가까운 단어 벡터는 '일본'이 되는 것이다. 이렇게 컴퓨터가 텍스트의 시맨틱을 "이해"할 수 있다는 장점으로 인해 텍스트 임베딩 모델에 관한 많은 연구가 진행되었는데, 다양한 방법들이 연구되다가 2013년 Mikolov가 획기적인 신경망(neural network) 기반 단어 벡터 임베딩 모델을 발표하면서 거의 대부분의 텍스트 임베딩 모델 연구들이 이 연구를 기반으로 비약적으로 발전하기 시작하였다.

1절에서는 대표적인 신경망 기반 텍스트 임베딩 모델 연구들을 소개하고, 2절에서는 단어를 뛰어넘어 문장 그리고 문단을 단위로 하는 대표적인 신경망 기반 임베딩 모델 연구를 소개하고자 한다.

2.2.1 단어 임베딩 모델 연구

신경망 기반 단어 임베딩 모델 연구들을 소개하기에 앞서 임베딩을 학습하는 과정에서 가장 중요한 가정에 대해 설명하겠다. Firth가 제시한 distributional hypothesis[13]가 바로 그것이다. Distributional hypothesis는 "***You shall know a word by the company it keeps.***" 라는 문장으로 설명되곤 하는데, 같이 등장하는 단어들이 비슷한 단어들은 서로 비슷한 의미를 가질 가능성이 많다, 즉 비슷한 문맥에 등장하는 단어들은 비슷한 뜻을 가진 단어일 확률이 높다는 것이다. 이 가정에 따라 단어들의 의미를 파악하려면 그 단어가 등장하는 문맥들의 규칙을 살펴봐야 하는데, 데이터의 양이 적다면 이러한 규칙들을 파악하기 힘들다. 따라서 대규모 데이터에 대해 문맥 규칙을 파악할 수 있는 신경망 기반 모델이 텍스트 임베딩 연구의 주류를 이루고 있다.

[14]는 단어 임베딩을 학습할 수 있은 신경망 기반 언어 모델(Neural Network Language Model, NNLM)을 최초로 제안하였다. 그림 2에서 볼 수 있듯이, NNLM은 인풋 레이어(input layer), 프로젝션 레이어(projection layer), 히든 레이어(hidden layer), 아웃풋 레이어(output layer) 총 네 개의 레이어(layer)로 이루어진 간단한 신경망이다. 학습 방법은 다음과 같다. 일단 현재 단어 w이전의 N개의 단어를 one-hot encoding으로 벡터화한다. One-hot encoding이란 길이가 사전의 크기 이며, 벡터화하고자 하는 단어 위치의 원소 값만 1이고 나머지는 0으로 이루어진 벡터이다. 프로젝션 레이어의 크기를 P라고 하였을 때, 각각의 벡터들은 $V \times P$ 크기의 프로젝션 행렬에 의해 곱해져 히든 레이어의 인풋으로 주어진다. 인풋으로 주어진 벡터들은 크기 H의 히든 레이어를 거쳐 아웃풋 레이어에서 길이 V의 벡터로 나오게 된다. 이 벡터의 i번째 원소 는 현재 단어가 w가 i번째 단어 일 확률이고, 이를 실제 단어 w의 one-hot encoding 벡터와 비교하여 에러를 계산, 오차역전파법(backpropagation)[16]을 거쳐 네트워크의 가중치(weight)값, 즉 프로젝션 행렬과 히든 행렬의 값들을 최적화 해나가게 된다. 최종적인 단어 벡터들은 프로젝션 행렬의 길이 P짜리의 행들이 된다.

[16] 오차역전파법(backpropagation)이란 아웃풋 레이어에서의 결과 벡터와 실제 원하는 결과 벡터의 에러를 계산하여, 이 에러를 각 레이어에 역전파시켜 각각 레이어의 에러 분포를 계산하고 이 에러를 이용하여 각 레이어의 가중치 값을 최적화하는 방법이다.

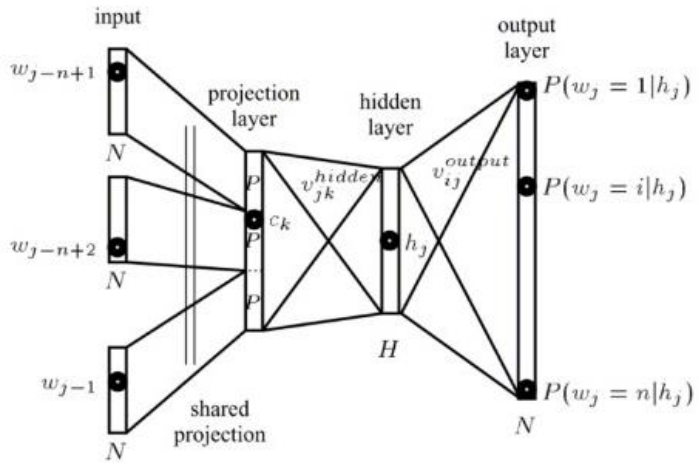

그림 2 Neural Network Language Model(NNLM)

이 모델은 단어의 벡터화라는 컨셉에 대한 새로운 지평을 열어주었고 현재 존재하는 모든 신경망 기반의 단어 임베딩 모델은 거의 이 모델의 컨셉을 이어 받아 발전시킨 모델이라고 할 수 있다. 하지만 이 모델은 앞 단어에 대한 문맥을 전혀 고려하고 있지 않고, 결정적으로 매우 느리다는 단점이 있다. 따라서 현재는 이를 개선한 여러 모델들이 나와 있고, 지금부터 설명할 Mikolov의 모델은 이 중 가장 대표적인 모델이라고 할 수 있다.

Mikolov의 연구[11]는 이른바 Word2Vec으로 불리며, [14]에서 제기된 두 가지 단점을 모두 해결한 단어 임베딩 모델을 제안하였다. 총 두 가지 모델을 제안하였는데, 하나는 CBOW(Continuous Bag-of-Words) 모델이고 다른 하나는 Skip-gram 모델이다.

CBOW 모델의 경우 [14]의 방법과는 다르게 현재 단어의 앞과 뒤의 단어들을 모두 고려한다. CBOW 모델은 인풋 레이어, 히든 레이어, 아웃

풋 레이어 총 세 개의 레이어로 구성되어 있다. 현재 단어의 앞과 뒤 각각 $\frac{C}{2}$개의 단어, 총 C개 단어들의 one-hot encoding 벡터가 히든 레이어의 인풋으로 들어가고, 히든 레이어의 사이즈가 $V \times N$ 행렬과 아웃풋 레이어의 사이즈가 $N \times V$인 행렬을 거쳐 아웃풋 레이어의 길이 V인 벡터로 나오게 된다. 이 벡터는 [14]의 모델과 마찬가지로 각 단어의 확률 값 벡터이며, 현재 단어의 one-hot encoding과 비교해 에러를 계산, 오차역전파법을 통해 행렬의 값들을 업데이트한다. 실제 단어 임베딩은 히든 레이어 행렬의 길이 N인 행들이다.

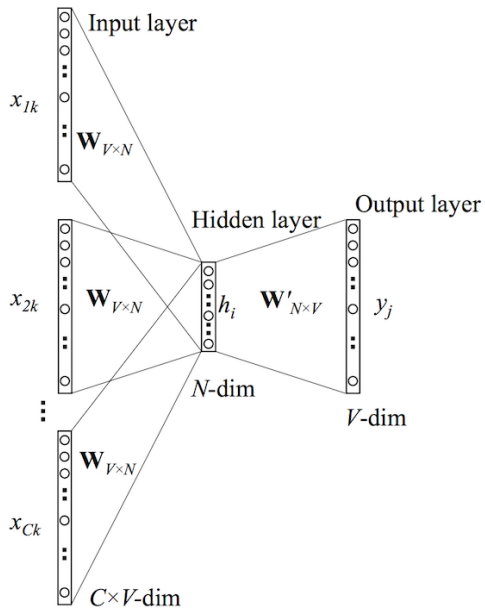

그림 3 Continuous Bag-of-Words(CBOW) 모델

Skip-gram 모델은 CBOW와 반대 방향의 모델이라고 할 수 있는데, 현재 주어진 단어 하나를 가지고 단어의 앞과 뒤에 등장하는 총 C개의 단

어를 유추하는 것이다. 기본적인 학습 방법과 구조는 CBOW와 반대 방향일 뿐 굉장히 유사하다. CBOW 모델과 Skip-gram 모델을 비교하면 CBOW 모델이 좀 더 논리적이고 사람의 직관에 더 잘 부합하지만, 실제 성능은 Skip-gram이 더 좋은 것으로 알려져 있다.

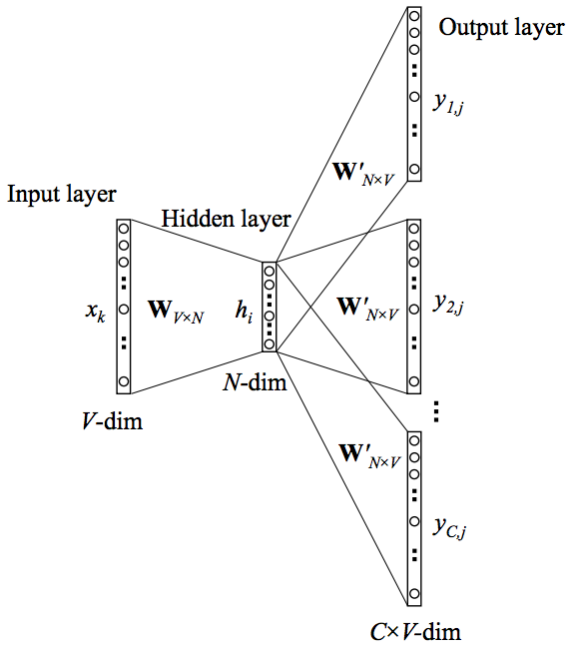

그림 4 Skip-gram 모델

2.2.2 문장 및 문서 임베딩 모델 연구

단어 임베딩 모델에 대한 연구가 진행됨에 따라 단어보다 더 큰 단위인 어구(phrase), 문장, 더 나아가서는 문서 전체를 벡터화 시키려는 연구들이 있었다. 특히 신경망 기반으로 학습한 단어 임베딩을 이용하여 더 큰 단위의 임베딩을 만들려는 시도들이 있었는데, 대부분 단어 임베딩을 평균 내거나, 단어에 TF-IDF 등의 가중치를 주어 가중치 평균을 내는 등

의 매우 단순한 방법에 지나지 않았다. 그러다 2014년 Le와 Mikolov의 연구가 나온 것을 시작으로 하여 신경망 기반으로 문장 및 문서 임베딩을 단독으로 학습하려는 연구들이 많이 등장하기 시작하였다.

[15]는 Paragraph vector라는 단어보다 큰 단위의 어떤 텍스트도 학습할 수 있는 신경망 기반 임베딩 모델을 제안하였다. 그림 5에서 볼 수 있듯이, 이 모델은 매우 간단한 구조로 이루어져 있다. 먼저 이 모델에는 두 개의 행렬이 존재하는데, 하나는 문단(paragraph) 행렬 D, 다른 하나는 단어 행렬 W이다. 단어 행렬 의 한 행은 단어 벡터이며, 문단 행렬 의 한 행은 문단 벡터이다. 즉, 문단 하나도 일종의 "단어" 역할을 하는 것이다. 한 문단에 대한 벡터를 학습할 때 문단 벡터와 그 문단에 속한 단어 벡터들은 두 번째 레이어에서 평균 되거나 연결되어 하나의 벡터가 되는데, 이 때 문단 벡터는 그 문단의 모든 문맥(context)들에 포함되어 같이 학습된다. 다시 말하면, 해당 문단 안의 특정 길이의 모든 단어 시퀀스(sequence)에 포함되어 같이 학습된다. 이렇게 하면 문단 벡터는 그 문단 안의 모든 문맥에 대한 정보를 포함할 수 있게 된다. 이것이 이 연구의 핵심 아이디어이다. 두 번째 레이어에서 하나의 벡터가 된 문맥 벡터는 해당 문맥의 다음 단어를 예측하고, 실제 다음 단어의 벡터와의 에러를 계산, 오차역전파법을 이용하여 모델의 파라미터(parameter) 값들을 최적화하게 된다.

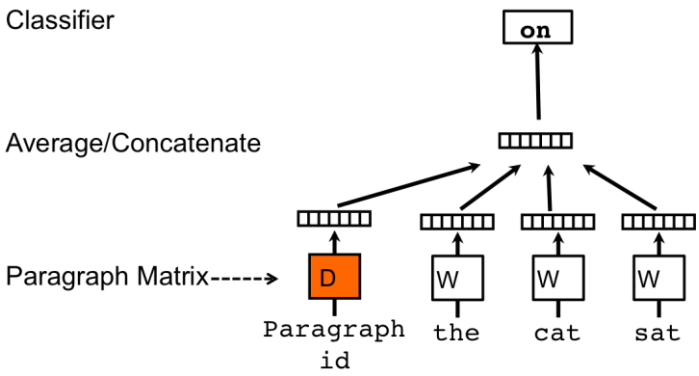

그림 5 Paragraph vector 모델

이 모델은 매우 간단함에도 불구하고 다양한 태스크에 대해 당시의 state-of-the-art 연구들보다 높은 성능을 보여주었으며, 현재까지도 다양한 문장 및 문서 임베딩 연구들의 베이스라인(baseline)으로 사용되고 있다. 3 장에서는 이 연구의 모델을 기반으로 하여 문장, 문서, 단어 임베딩을 학습하기 위한 임베딩 모델을 제안한다.

제 3장 텍스트 임베딩을 이용한 퀴즈 생성

3장에서는 텍스트 임베딩을 사용한 새로운 객관식 빈칸 채우기 퀴즈 자동 생성 방법을 제안한다. 1절에서는 기존의 객관식 빈칸 채우기 퀴즈 자동 생성 연구의 한계점을 살펴보고, 2절에서는 이를 해결하기 위하여 단어, 문장, 문서의 임베딩을 동시에 학습할 수 있는 새로운 임베딩 모델을 제안한다. 그리고 3절에서는 학습된 임베딩을 이용하여 객관식 빈칸 채우기 퀴즈를 생성하는 유사도 기반 퀴즈 생성 방법을 제안한다. 마지막으로 4절에서는 최종적으로 나오는 퀴즈의 질을 향상시킬 수 있는 키워드 기반 사후 필터링 방법을 소개한다. 그림 6은 본 논문의 퀴즈 생성 프로세스(process)를 나타낸 것이다.

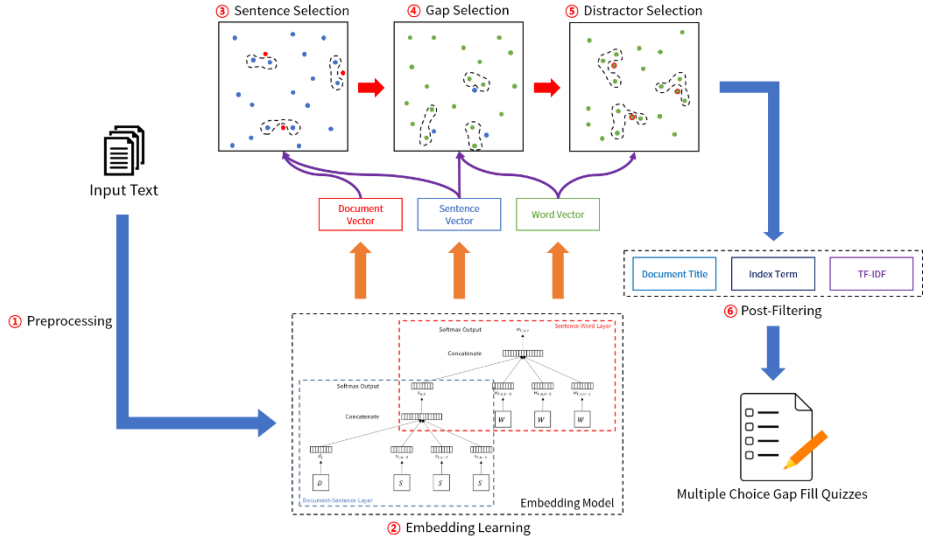

그림 6 객관식 빈칸 채우기 퀴즈 생성 프로세스

3.1 기존 연구의 한계점

기존 객관식 빈칸 채우기 퀴즈 연구들은 도메인에 종속적인 룰과 피쳐들을 사용하였다. 즉, 각 연구들은 그들이 사용하는 텍스트의 도메인 및 종류에 최적화된, 사람이 직접 생성한 룰과 피쳐들을 이용하여 퀴즈를 생성하였다. [6]은 영어 어휘와 문법 퀴즈 만을 생성하기 위한 룰을 사용하였고, [7]은 자신들의 연구에 쓰인 생물학 책에 적합한 피쳐를 직접 만들어내어 이를 스코어링에 이용, 퀴즈를 생성하였다. [8]은 위키피디아 문서에 적합한 피쳐를 직접 만들어내어 이를 분류기의 피쳐로서 사용하였다. 특히 이들이 사용한 위키피디아 링크(link) 피쳐 등은 위키피디아 문서에 대해서만 사용할 수 있는 피쳐이기 때문에 다른 도메인에는 이 연구의 방법을 적용할 수 없다. [10]은 이러한 도메인 종속적인 피쳐를 다른 연구에 비해서는 덜 사용하긴 했지만 여전히 분류 방법에 이러한 피쳐들을 사용하고 있다.

도메인 종속적인 룰과 피쳐의 문제점은 이러한 피쳐들이 다른 도메인의 텍스트에는 범용적으로 쓰이기 어렵다는 것이다. 그리고 쓰이더라도 특정 도메인에 최적화되어있기 때문에 생성되는 퀴즈의 질은 그 도메인에서보다는 떨어질 수밖에 없다. 이를 위해서는 사람이 만드는 피쳐의 개수와 영향력을 최대한 줄여야 하지만, 기존 연구들의 퀴즈 생성 방법들, 즉 피쳐 기반 스코어링, 그리고 분류 방법 등은 상당히 많은 피쳐가 필요하기 때문에 퀴즈 생성 방법을 근본적으로 바꾸지 않는 이상 이러한 피쳐는 불가피하게 쓰일 수밖에 없다.

또 다른 한계점은 텍스트의 시맨틱을 중요시하고 있지 않다는 점이다. 기존의 객관식 퀴즈 자동 생성 연구들은 각 단계, 즉 문장, 갭, 오답지 선택 단계에서 사람이 직접 만들어낸 룰과 피쳐들을 사용하였다. 이러한 룰과 피쳐들은 퀴즈 생성에 대한 사람의 직관을 반영한 것이다. 즉 사람이 퀴즈 생성 측면에서 좋은 문장, 갭, 오답지의 특성을 생각하여 룰과 피쳐를 만들어낸 것이다. 하지만 이러한 룰과 피쳐들은 텍스트의 구조적 특성에서 나온 것이 대부분이다. 예를 들면, 기존 연구들은 문장 선택 단계에서는 문서에서의 문장의 위치, 그리고 특정 단어들의 등장 빈도 등을, 갭 선택 단계에서는 특정 품사 또는 등장 빈도 등을 피쳐로 삼았다. 오답지 선택 단계에서는 갭과의 유사도를 사용하긴 했지만 이 유사도도 갭과 오답지 후보의 주변에 얼마나 같은 단어가 많이 등장하는지, 즉 카운트(count) 기반의 유사도 척도를 사용하였다. 하지만 사람이 퀴즈를 만들 때에는 이러한 구조적 특성에는 잘 주목하지 않는다. 오히려 어떤 문장이나 갭이 이 문서에서 중요한 의미를 가지는 문장, 갭인지, 오답지는 갭과 얼마나 의미적으로 비슷한 지를 고려하여 퀴즈를 만든다. 이러한 점을 고려했을 때, 자동 퀴즈 생성 문제에서 텍스트의 시맨틱을 파악하는 것은 매우 중요하고 반드시 필요하다고 할 수 있다.

2절과 3절에서는 이러한 객관식 퀴즈 자동 생성 연구의 문제점들을 해결할 수 있는 새로운 임베딩 모델 및 퀴즈 생성 방법을 제시한다.

3.2 문서, 문장, 단어 학습 임베딩 모델

1절에서 살펴본 바 텍스트의 시맨틱은 퀴즈 생성 문제에 있어서 매우

중요하다는 것을 알 수 있었다. 따라서 이 절에서는 객관식 퀴즈 생성 측면에서 문서, 문장, 단어의 시맨틱을 모두 파악할 수 있는, 즉 문서, 문장, 단어의 임베딩을 함께 학습할 수 있는 새로운 임베딩 모델을 제안하고자 한다.

먼저 가장 쉽게 생각해볼 수 있는 임베딩 모델은 단어 임베딩을 평균 내어 문장 임베딩을 만들고, 문장 임베딩을 평균 내어 문서 임베딩을 만드는 **평균 모델(average model)**이다. 먼저 [11]의 Skip-gram 등의 모델을 이용하여 입력 문서에 대해 단어 임베딩을 학습한다. 그리고 한 문장에 등장하는 모든 단어들의 임베딩을 평균하여 문장 임베딩을 만들고, 한 문서에 등장하는 모든 문장들의 임베딩을 평균하여 문단 임베딩을 만든다. 이 모델은 가장 쉽게 문장 및 문서 임베딩을 같이 생성할 수 있는 모델이지만, 평균을 내기 때문에 단어 및 문장의 "순서" 정보가 유실된다는 단점이 있다. 즉, "***Mike loves Catherine.***" 이라는 문장과 "***Catherine loves Mike.***" 라는 문장의 임베딩이 이 모델에서는 같은 임베딩을 갖게 되는 것이다. 이렇게 단어 임베딩을 평균 내어 문장 및 문서 임베딩을 만드는 시도들은 기존에도 있어왔지만, 위와 같이 순서 정보가 유실된다는 단점이 지적되어 현재는 베이스라인으로만 쓰이고 있는 모델이다.

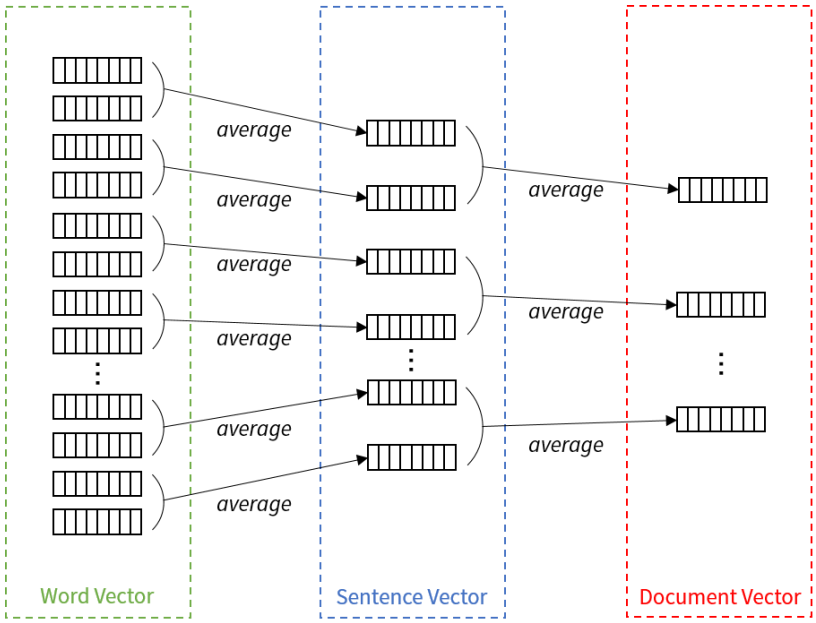

그림 7 Average 모델

그림 8은 새로 제안하는 임베딩 모델의 구조이다. 이 임베딩 모델은 [15]의 Paragraph vector 모델의 아이디어를 기반으로 하되, Paragraph vector 모델은 문단 임베딩과 단어 임베딩, 즉 두 가지의 임베딩을 동시에 학습할 수 있었던 반면 새로 제안하는 **합동 모델(joint model)**은 문서, 문장 그리고 단어 임베딩, 총 세 가지의 임베딩을 학습할 수 있다. 또한 모델은 평균 모델에서처럼 순서 정보가 유실되는 문제점도 없다.

이 임베딩 모델은 크게 두 부분으로 나눌 수 있다. 첫 번째는 오른쪽 상단 박스로 표시되어 있는 문장-단어 레이어(Sentence-Word Layer)이고, 두 번째는 왼쪽 하단 박스로 표시되어 있는 문서-문장 레이어(Document-Sentence Layer)이다. 이 모델에는 총 세 개의 행렬, 문서 행렬 D, 문장 행렬 S, 그리고 단어 행렬 W가 존재한다. 문장-단어 레이어에서는 단어 임

베딩과 문장 임베딩이 학습되고, 문서-문장 레이어에서는 문장 임베딩과 문서 임베딩이 학습된다. 특히 문장을 문서를 구성하는 하나의 "단어"로 취급하여 문서 임베딩이 문장 임베딩으로부터 학습될 수 있도록 하였다. 문서-문장 레이어에서는 문서 임베딩과 문장 임베딩들이 연결되어 다음 문장을 예측하고, 문장-단어 레이어에서는 문장 임베딩과 단어 임베딩들이 연결되어 다음 단어를 예측한다. 그리고 이 과정에서의 에러는 단어 행렬, 문장 행렬, 그리고 문서 행렬에 전파되어 이들 행렬의 값들이 업데이트 되게 된다. 새로운 임베딩 모델은 문서, 문장, 단어 임베딩을 동시에 학습한다는 면에서 합동 모델로 칭하였다.

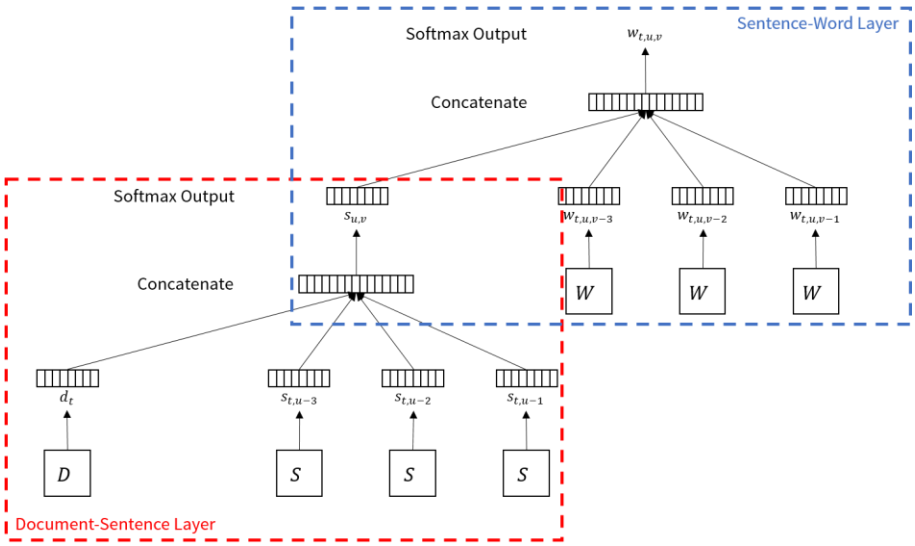

그림 8 Joint 모델

3절에서는 이러한 합동 모델로 학습된 문서, 문장, 단어 임베딩을 사용하여 퀴즈를 생성하는 유사도 기반 퀴즈 생성 방법을 제시한다.

3.3 유사도 기반 퀴즈 생성 방법

사람이 어떤 문서에 대해 빈칸 채우기 퀴즈를 만들 때에는 어떤 문장이 이 문서에서 가장 중요한지, 그리고 선택된 문장에서 어떤 단어가 가장 중요한지를 고려한다. 이 때 가장 중요하다는 것은 해당 문서 및 문장을 가장 잘 핵심적으로 요약한다는 것이다. 요약한다는 것은 결국 문서 및 문장이 나타내고 있는 의미를 함축시켜 표현한다는 것이기 때문에 요약의 결과는 요약의 대상이 되는 글의 의미와 필연적으로 비슷하게 된다. 이에 착안한 그래프 기반 자동 요약 연구들은 다른 텍스트들과 가까운 거리에 있는 텍스트들을 중요한 텍스트로 보고 이를 선택하여 요약을 완성한다.[5, 16]

본 논문은 이러한 사실에 착안하여 유사도 기반 빈칸 채우기 퀴즈 생성 방법을 제안하고자 한다. 방법은 간단하다. 상위 레벨의 텍스트를 가장 잘 "요약"하는 하위 레벨의 텍스트를 퀴즈의 대상으로 선택하는 것이다. 여기서 잘 "요약"한다는 것은 의미가 비슷하다는 이야기와 일맥상통하므로, 결국 상위 레벨의 텍스트와 의미가 비슷한 하위 레벨의 텍스트를 선택하는 것이다. 유사도 기반 빈칸 채우기 퀴즈 생성 방법은 알고리즘 1에 나타나있다.

Algorithm 1 Similarity-Based MC GF Quiz Generation

Input: Document List D, Embedding List E, Vocabulary V

# of sentences to select per a document k_{ds}

# of gaps to select per a sentence k_{sg}

# of distractors to select per a gap k_{gt}

Output: Multiple-Choice Gap-Fill Quiz List Q

$Q \leftarrow [\,]$

for d **in** D

    for s **in** sent_tokenize(d)

        Calculate dist($E[d], E[s]$)

    $S_d \leftarrow$ top-k_{ds} closest sentences to the document d

    for s **in** S_d

        for g **in** word_tokenize(s)

            Calculate dist($E[s], E[g]$)

        $G_s \leftarrow$ top-k_{sg} closest gaps to the sentence s

        for g **in** G_s

            for t **in** V

                Calculate dist($E[g], E[t]$)

            $T_g \leftarrow$ top-k_{gt} closest distractors to the gap g

            $q \leftarrow (s, g, T_g)$

            Q.append(q)

return Q

퀴즈 생성에 앞서 2절에서 설명한 임베딩 모델로 문서, 문장, 단어 임베딩 E를 학습한다. 그 후 문장 선택 단계에서 문서를 잘 요약하는, 즉 문서와 가장 가까운 k_{ds}개의 문장들을 퀴즈의 대상으로 선택한다. 2장 1절에서 설명했듯이 텍스트 임베딩은 텍스트의 시맨틱을 표현하기 때문에 임베딩 사이의 거리가 가까우면 의미가 비슷하다. 다만 선택되는 문장들은 당연히 그 문서에 속한 문장들이어야 한다. 그 후 갭 선택 단계에서는 문장 선택 단계에서 선택된 문장과 가까운 k_{sg}개의 단어들을 갭으로 선택한다. 역시 문장과 가까운 단어들이 해당 문장을 잘 요약할 것임을 나타낸다. 다만 갭들은 명사와 수(cardinal)여야 한다. 이는 중요한 단어들이 거의 대부분 명사이고, 수들도 연도 등 중요한 경우가 많다는 경험에 의한 분석에 따른 것이다. 형용사나 부사 등은 갭에 고려하지 않았다.

오답지 선택 단계도 마찬가지로 유사도 기반 방법으로 이루어진다. 갭 선택 단계에서 선택된 갭들과 가까운 k_{gt}개의 단어들을 오답지로 선택한다. 다만 문서 내의 서로 다른 단어의 수, 즉 사전의 크기 $|V|$가 너무 크기 때문에 유사도만 이용해서는 오답지들의 품질을 보장하기 어렵다고 판단하여 몇 가지의 추가적인 기준을 두었다.

첫째, 오답지는 갭과 같은 퀴즈 문장에 있어선 안 된다. 오답지가 퀴즈 문장에 있다면 이는 명백한 오답이 되어 오답지로서의 기능을 하지 못할 것이다.

둘째, 갭과 오답지는 서로 같은 품사 여야 한다. 즉, 갭이 명사이면 오답지도 명사 여야 하고, 갭이 수이면 오답지도 수여야 한다. 갭과 오답지가 서로 품사가 다르다면 역시 이 오답지는 명백한 오답이 될 것이다.

셋째, 갭과 오답지는 서로 같은 의미적 카테고리에 있어야 한다. 이 때의 의미적 카테고리라 함은 WordNet의 lexicographer file의 이름(lexname) 45개를 말한다. 이 의미적 카테고리는 형용사, 명사, 동사, 부사 등을 의미에 따라 분류한 것인데, noun.person, noun.phenomenon, verb.emotion, verb.social과 같은 카테고리들이 있다. 이러한 조건을 둔 것은, 두 단어가 최소한 큰 틀에서 같은 의미의 카테고리에 속해야 최소한의 오답지의 퀄리티를 보장할 수 있다는 것으로, 매우 약한 제약 조건이라고 할 수 있다.

넷째, 갭과 오답지가 서로 동의어이면 안 된다. 동의어(synonym)란 단어는 다르지만 뜻이 같거나 비슷한 단어들을 말하는데, 예를 들어 "아기"와 "유아", "구입"과 "구매", "가게"와 "상점" 등이 있다. 갭과의 동의어인 오답지는 또 다른 정답으로 해석될 여지가 많다. 따라서 갭과 동의어인 오답지는 제외하는데, 이를 위해 WordNet의 synset을 이용하였다. Synset은 WordNet 안에서 서로 동의어 관계에 있는 단어들의 집합인데, 갭과 같은 synset에 포함되는 단어들은 오답지에서 제외하였다.

위의 네 가지 기준은 오답지로서 최소한의 기준에 미달하는 오답지를 거르기 위한 기준이며, 오답지가 가져야 하는 "최소한"의 자격 조건을 선정한 것이기 때문에 유사도 기반 오답지 선택 방법을 이용하여 나오는 오답지 중 매우 일부만 이 기준에 해당한다. 따라서 유사도 기반 오답지 선택 방법이 오답지 선택 단계의 핵심이며, 위의 네 가지 조건은 부차적인 것이라고 할 수 있다.

이 절에서는 유사도 기반 객관식 빈칸 채우기 퀴즈 생성 방법을 제안

하였다. 이 알고리즘은 어떠한 외부의 사전, 즉 미리 정의된 중요한 키워드 리스트 없이도 퀴즈를 생성할 수 있기 때문에 다양한 텍스트에 대해 범용으로 사용할 수 있다는 장점이 있다. 하지만 어떤 텍스트의 경우에는 미리 정의된 중요한 키워드 리스트가 있는 경우가 있다. 바로 교과서에 나오는 색인(index)이 그것이다. 색인에는 해당 교과서에서의 중요한 단어가 존재하는데, 이러한 중요한 단어들은 퀴즈 생성 관점에서 굉장히 중요한 도움을 줄 수 있다. 본 논문은 이러한 사실에 착안하여 4절에서는 키워드를 이용하여 최종적으로 생산되는 퀴즈의 질을 대폭 끌어올릴 수 있는 키워드 기반 사후 필터링 방법을 제안한다.

3.4 키워드 기반 사후 필터링 방법

앞서 설명한 대로 교과서와 같은 특정 텍스트는 색인과 같은 중요한 키워드들의 리스트가 존재한다. 이러한 키워드들은 갭이나 오답지의 좋은 후보가 될 수 있을 뿐만 아니라 퀴즈 생성 측면에서의 중요한 문장의 경우 이러한 키워드들이 많이 포함되어 있을 확률이 많아 퀴즈 생성에서의 중요한 힌트가 될 수 있다. 따라서 어떤 텍스트에 대해 이러한 중요한 키워드 리스트가 있다면 최종적으로 생산되는 퀴즈의 질을 향상시키기 위하여 사용하는 것이 바람직하다고 할 것이다.

이 절에서는 3절의 알고리즘을 통해 기계적으로 생산된 다량의 퀴즈를 미리 정의된 키워드 리스트를 이용하여 필터링하는, 즉 질이 나쁜 퀴즈들을 제외시키는 사후 필터링(post-filtering) 방법을 제안한다. 또한 미리 정의된 키워드 리스트가 존재하지 않는 텍스트의 경우를 위하여 문서의

제목과 TF-IDF값도 이용한다.

먼저 미리 정의된 키워드 리스트에 대한 문장 및 단어 점수 $ScoreK_s$와 $ScoreK_w$를 정의한다.

$$ScoreK_s(d,s) = \frac{|s \cap K|}{|d \cap K|} \tag{3.1}$$

$$ScoreK_w(w) = \begin{cases} 1, & w \in K \\ 0, & w \notin K \end{cases} \tag{3.2}$$

위의 두 식에서 d, s, w는 각각 문서, 문장, 그리고 단어를 의미하며, K는 미리 정의된 키워드 집합을 의미한다. $ScoreK_s$는 문서 d에 포함된 키워드의 개수와 문장 s에 포함된 키워드 개수의 비율로, 문장 s가 문서 d에 포함된 키워드들 중 어떤 비율만큼의 키워드를 가지고 있는지를 나타낸 것이다. 그리고 $ScoreK_w$는 단어 w가 키워드인지 아닌지를 불린(Boolean)으로 나타낸 것이다. 위의 두 점수를 이용하여 객관식 빈칸 채우기 퀴즈의 키워드 리스트에 대한 점수 $ScoreK_q$를 다음과 같이 정의할 수 있다.

$$\begin{aligned} &ScoreK_q(d,s,g,T) \\ &= w_s ScoreK_s(d,s) + w_g ScoreK_w(g) \\ &+ \frac{w_T}{|T|}\sum_{t \in T} ScoreK_w(t) \end{aligned} \tag{3.3}$$

$ScoreK_q$는 키워드 리스트 측면에서 퀴즈의 각 요소, 즉 문장, 갭, 그리

고 오답지의 질을 나타내는 점수이다. 위의 식에서 d, s, g, T는 각각 문서, 문장, 갭, 그리고 오답지 집합을 나타낸다. $ScoreK_s(d,s)$는 퀴즈 문장에 대한 점수이며, $ScoreK_w(g)$ 는 갭에 대한 점수, 그리고 $\frac{w_T}{|T|}\sum_{t\in T} ScoreK_w(t)$는 오답지에 대한 점수이다. 그리고 각각의 점수에 가중치 w_s, w_g, w_T를 두어 각 요소들의 중요도를 조절할 수 있도록 하였다. 단, $w_s + w_g + w_T = 1$이다.

또한 미리 정의된 키워드 리스트가 없을 경우를 위하여 문서의 제목에 포함된 단어에 대한 점수를 정의하였다. 문서의 제목에는 그 문서가 설명하고자 하는 핵심 개념이나 단어들이 포함되어있기 때문에 이 또한 키워드 리스트와 비슷하게 중요한 역할을 할 수 있다. 먼저 키워드 리스트에서와 동일하게 문장 및 단어 점수 $ScoreL_s$와 $ScoreL_w$를 정의한다.

$$ScoreL_s(d,s) = \frac{|s \cap title(d)|}{|title(d)|} \tag{3.4}$$

$$ScoreL_w(d,w) = \begin{cases} 1, & w \in title(d) \\ 0, & w \notin title(d) \end{cases} \tag{3.5}$$

$ScoreL_s$는 문장 s에 문서 d의 제목 $title(d)$에 포함된 단어들이 얼마나 많은지 그 비율을 나타내는 점수이고, $ScoreL_w$는 단어 w가 문서 d의 제목에 있는지를 불린으로 나타낸 것이다. 위의 두 점수를 이용하여 객관식 빈칸 채우기 퀴즈의 문서 제목에 대한 점수 $ScoreL_q$를 다음과 같이 정의할 수 있다.

$$
\begin{aligned}
&ScoreL_q(d,s,g,T) \\
&= w_s ScoreL_s(d,s) + w_g ScoreL_w(g) \\
&+ \frac{w_T}{|T|}\sum_{t\in T} ScoreL_w(t)
\end{aligned}
\tag{3.6}
$$

$ScoreL_q$는 문서 제목 측면에서 퀴즈의 각 요소가 얼마나 질이 좋은 것인지를 나타내는 점수이다. 식의 구성은 식 3.3과 같다.

키워드 리스트와 문서 제목 점수는 기본적으로 중요한 키워드가 객관식 퀴즈에 얼마나 존재하는지를 점수로 환산한 것이다. 하지만 여기에 각 키워드의 중요도는 반영되어 있지 않다. 예를 들어 세계사 책에서 "전쟁"이라는 단어와 "장미전쟁"이라는 단어는 모두 키워드이지만 "장미전쟁"과 달리 "전쟁"이라는 단어는 세계사 책에서 매우 흔하기 때문에 "장미전쟁"이 조금 더 중요한 키워드라고 할 수 가 있다. 따라서 이러한 키워드들의 중요도를 반영하기 위해 TF-IDF 점수를 도입하였다. TF-IDF(Term Frequency-Inverse Document Frequency)란 정보 검색(Information Retrieval)과 텍스트 마이닝(Text Mining) 등에서 주로 이용하는 점수로, 여러 문서로 이루어진 문서 집합이 있을 때 어떤 단어가 특정 문서 내에서 얼마나 중요한 것인지를 나타내는 수치이다. TF-IDF의 식은 다음과 같다.

$$
\begin{aligned}
TF-IDF(t,d) &= TF(d,t)\cdot IDF(t) \\
&= \log(1+\frac{n(d,t)}{n(d)})\cdot\frac{1}{n(t)}
\end{aligned}
\tag{3.7}
$$

$TF(d,t)$는 단어 t가 문서 d에서 얼마나 자주 등장하는지를 나타내는 점수로, $n(d)$는 문서 d의 단어의 개수, $n(d,t)$는 문서 d에 등장하는 단어 t의 개수이다. $IDF(t)$는 단어 t가 다른 문서에 얼마나 등장하는지를 나타내는 점수로, $n(t)$는 단어 t가 등장하는 문서의 개수이다. 즉, 단어의 문서 내의 등장 빈도와 그 단어가 다른 문서에 흔히 등장하는 단어인지를 모두 고려하여 단어의 중요도를 계산한다. 이를 이용하여 문장 및 단어 점수 $TF-IDF_s$와 $TF-IDF_w$를 정의한다.

$$TF-IDF_s(d,s)=\frac{1}{|s|}\sum_{w\in s}TF-IDF(w,d) \tag{3.8}$$

$$TF-IDF_w(d,w)=TF-IDF(w,d) \tag{3.9}$$

단어의 $TF-IDF$ 점수 $TF-IDF_w$는 식 3.7과 같은 값이며 문장의 $TF-IDF$ 점수 $TF-IDF_s$는 문장 내의 모든 단어의 값을 평균한 것이다. 이에 기반하여 객관식 빈칸 채우기 퀴즈의 $TF-IDF$에 대한 점수 $TF-IDF_q$를 다음과 같이 정의할 수 있다.

$$\begin{aligned}&TF-IDF_q(d,s,g,T)\\&=w_sTF-IDF_s(d,s)+w_gTF-IDF_w(d,w)\\&+\frac{w_T}{|T|}\sum_{t\in T}TF-IDF_w(d,t)\end{aligned} \tag{3.10}$$

이렇게 세 요소에 대해 정의한 퀴즈 점수를 모두 합한 최종적인 퀴즈

점수는 다음과 같이 정의한다.

$$ScoreQuiz_f(q) = w_K ScoreK_q(q) + w_L ScoreL_q(q) + w_{TF-IDF} TF - IDF_q(q) \quad (3.11)$$

이렇게 정의된 퀴즈 점수 $ScoreQuiz_f(q)$는 3절의 알고리즘을 통해 대량으로 생성된 퀴즈의 사후 필터링에 사용된다. 각 퀴즈의 스코어를 계산한 후 일정 스코어에 미치지 못하는 퀴즈는 모두 버려진다. 하지만 이렇게 퀴즈가 버려진다 하더라도 3절의 알고리즘을 통해 매우 많은 퀴즈가 생성되기 때문에 필터링 이후 퀴즈의 수도 많은 수준이며, 최종적으로 나오는 퀴즈의 품질이 높은 것이 중요하기 때문에 이러한 사후 필터링 방법이 유효하다고 하겠다.

4장에서는 기존 연구의 퀴즈 품질과 3절의 알고리즘으로 생성한 퀴즈의 품질을 비교하고, 여기에 사후 필터링을 거친 퀴즈의 품질도 비교하여 본 논문에서 제안한 유사도 기반 퀴즈 생성 알고리즘과 키워드 기반 사후 필터링 방법의 우수성을 함께 보이겠다.

제 4장 실험 방법 및 결과

4장에서는 본 논문에서 제안한 퀴즈 생성 방법 및 사후 필터링 방법의 성능을 평가한다. 1절에서는 실험 방법을 소개하고, 2절에서는 실험 결과와 이에 대한 성능 평가를 서술한다.

4.1 실험 방법

4.1.1 데이터 셋

실험에는 총 두 가지의 영어 전공 책을 이용하였다. 하나는 대표적인 데이터베이스 전공 서적 중 하나인 **Database System Concepts 6th edition**[17]이고, 다른 하나는 대표적인 생물학 교과서로 불리는 **Campbell Biology 9th edition**[18]이다. 이 두 책에 대해 텍스트 만을 추출한 후, 문서 단위로 토큰화(tokenize)한다. 이 때 문서의 단위는 가장 레벨이 낮은 섹션 단위로 하였는데, 그 이유는 문서가 너무 길어지는 것, 즉 문서 안의 문장의 수가 너무 많아지는 것을 방지하고 가급적이면 문서 안의 모든 텍스트가 같은 소주제에 대한 내용을 설명하고 있는 것이 질 좋은 임베딩의 학습에 유리하다고 판단하였기 때문이다. 책을 문서 단위로 토큰화한 다음에는 다시 문장 단위로, 그리고 단어 단위로 토큰화하여 데이터 셋을 구축한다. 또한 각 책의 색인에 있는 단어들을 추출하였다. 두 전공 책에 대한 데이터 셋 통계는 표 2과 같다.

표 2 데이터 셋 통계

	데이터베이스 책	생물학 책
페이지 수	1,376	1,472
챕터 수	30	56
문서 수	982	1,761
문장 수	19,613	24,245
단어 수	592,225	751,063
사전 크기 (서로 다른 단어 수)	9,139	18,226
색인 단어 수	1,931	8,232

기본적으로 데이터베이스 책보다 생물학 책이 전체적인 텍스트의 양이 많았는데, 그 이유는 데이터베이스 책은 하나의 컬럼(column)으로 서술된 데 반해 생물학 책은 두 개의 컬럼으로 서술되었기 때문이다. 더 주목할 만한 것은 사전 크기와 색인 단어 수인데, 사전 크기는 생물학 책이 2배, 색인 단어 수는 무려 4배 이상의 차이를 보였다. 아무래도 생물학의 특성상 다양한 용어들이 많이 등장하기 때문에 이렇게 큰 차이를 보인다고 할 수 있다.

4.1.2 임베딩 학습

1.1절에서 구축된 데이터 셋에 대해 본 논문에서 제시된 임베딩 모델을 이용하여 임베딩을 학습하였다. GPU는 NVIDIA GeForce GTX 980Ti를

사용하였으며, 구현 언어는 Python, 딥 러닝(deep learning) 라이브러리(library)로는 Chainer[17]를 사용하였다. 벡터의 유닛(unit), 즉 벡터의 길이는 200으로 하였으며, 단어 윈도우(window)의 크기는 5, 문장 윈도우의 크기는 3으로 하였다. 배치(batch)의 크기는 1000으로 하였으며, 에포크(epoch), 즉 학습 반복 횟수는 20으로 하였다. 그리고 단어 임베딩에 대해서는 미리 학습된 임베딩 값을 초기화 용도로 사용하였는데, 단어 임베딩 모델 중 하나인 GloVe[19]의 미리 학습된 200 유닛의 단어 임베딩을 사용하였다.[18]

4.1.3 평가 기준

기존 객관식 빈칸 채우기 퀴즈 연구들은 퀴즈의 질을 세 가지 측면에서 평가하였다. 퀴즈 문장의 질, 갭의 질, 그리고 오답지들의 질을 사람의 손을 빌려 각각 따로 평가하였다. 본 논문에서도 퀴즈의 질을 위에서 언급한 세 가지 측면에서 사람의 노력으로 평가하고자 하였다. 다만 각 요소의 품질을 평가할 때 명확한 기준이 있어야 더욱 객관적이고 정확한 평가가 이루어질 것이다. 본 논문에서는 표 3와 같은 평가 기준을 세워 문장, 갭, 오답지들의 질을 평가하였다.

[17] http://chainer.org/

[18] https://nlp.stanford.edu/projects/glove/

표 3 평가 기준

	평가 기준
문장	특정 개념에 대해 정의하고 있는 문장인가?
	특정 사실에 대해 설명하고 있는 문장인가?
	문장 내에서 찾을 수 없는 정보에 대해 설명하고 있진 않은가?
갭	해당 문장에서 중심이 되는 핵심 단어인가?
	다른 문서나 문장에서도 많이 등장하는 너무 일반적인 단어는 아닌가?
오답지	갭과 비슷한 상황에서나 용도로서 쓰이는 단어인가?
	실제로 문제를 푸는 사람을 헷갈리게 할 수 있을만한 단어인가?

4.1.4 실험 세팅

먼저 1.1절에서 언급한 데이터베이스 책과 생물학 책에 대하여 1.2절의 방법으로 임베딩을 학습한 다음 본 논문에서 소개한 유사도 기반 퀴즈 생성 방법을 이용하여 퀴즈를 대량으로 생성하였다. 각 문서에 대해 상위 4개의 문장을 퀴즈 문장으로 선택하였고, 문장당 상위 2개의 단어를 갭으로 선택하였다. 그리고 갭당 3개의 오답지를 생성하여 보기를 4개 가진 객관식 빈칸 채우기 퀴즈를 생성하였다. 이렇게 생성된 총 퀴즈의 개수는 데이터베이스 책이 7,263개, 생물학 책이 13,136개였다. 그리고 생성된 퀴즈 중 각각의 책에서 퀴즈의 점수가 가장 높은 500개의 퀴즈를

선택하였다. 이 때의 퀴즈 점수는 3.4절에서 언급한 필터링에서의 점수와는 다른 것으로, 유사도 기반 퀴즈 생성 알고리즘에서 이용하는 유사도에 기반한 점수이다. 그 식은 다음과 같다.

$$ScoreQuiz_s(d,s,g,T) = 1 - (w_s \text{dist}(d,s) + w_g \text{dist}(s,g) + \frac{w_T}{|T|}\sum_{t\in T} \text{dist}(g,t)) \quad (4.1)$$

이 점수는 퀴즈 생성 과정에서 쓰이는 유사도를 점수화한 것으로, 문서와 문장, 문장과 갭, 그리고 갭과 오답지 간의 유사도가 높을수록 점수가 높아진다. 즉, 본 논문에서 제안하는 퀴즈 생성 알고리즘에서 가장 좋다고 평가되는 퀴즈를 선택하기 위한 점수인 것이다. 이 점수를 이용하여 각각의 책에서 뽑은 500개의 퀴즈 중 50개를 랜덤으로 선택하였다. 또한 본 논문에서 소개한 키워드 기반 사후 필터링 방법의 성능을 평가하기 위하여 데이터베이스 책에서 나온 7,263개, 생물학 책에서 나온 13,136개 퀴즈에 대해 0.3의 기준 점수(threshold)를 두어 각각 2,564개, 937개의 퀴즈를 뽑아낸 후 이중 각각 50개를 랜덤으로 선택하였다. 이 때의 가중치 값은 $(w_s, w_g, w_D) = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$으로 모두 같게 하였다. 그리고 기존 객관식 빈칸 채우기 퀴즈 연구와의 성능 비교를 위해 각각의 책에 대하여 [7]에 의해 생성된 퀴즈들 중 역시 50개를 랜덤으로 선택하였다. 즉, 각 책마다 본 논문의 유사도 기반 퀴즈 생성 방법으로 생성한 퀴즈 50개, 본 논문의 키워드 기반 사후 필터링 방법을 거친 퀴즈 50개, 그리고 [7]의 방법으로 생성된 퀴즈 50개, 총 150개의 퀴즈(두 책을 합치면

총 300개)를 평가 대상으로 삼았다.

평가자가 해당 책에 대한 지식이 어느 정도 있어야 더욱 정확하고 합리적인 평가가 가능하다고 판단하여 컴퓨터공학 및 생물학 전공자 총 10명을 평가자로 선정하였다. 5명의 컴퓨터공학 전공자가 데이터베이스 퀴즈 150개를, 5명의 생물학 전공자가 생물학 퀴즈 150개를 각각 평가하였다. 평가할 때의 기준은 표 3를 따랐으며, 문장, 갭, 그리고 오답지에 대해 각각 따로 평가하였다. 문장과 갭에 대해서는 평가 기준을 만족하는지에 대해 좋음(1), 나쁨(0) 두 단계의 점수로 평가하였으며, 오답지에 대해서는 오답지 각각에 대해 좋음(1), 나쁨(0) 두 단계의 점수로 평가하여 좋음으로 평가한 오답지의 개수를 답하게 하였다.

4.2 실험 결과

10명의 평가자에게 평가를 진행하여 그림 9, 그림 10과 같은 결과를 얻었다. 각 점수는 평가자들의 점수를 평균한 것이며, 퀴즈 하나 당 평균값이다.

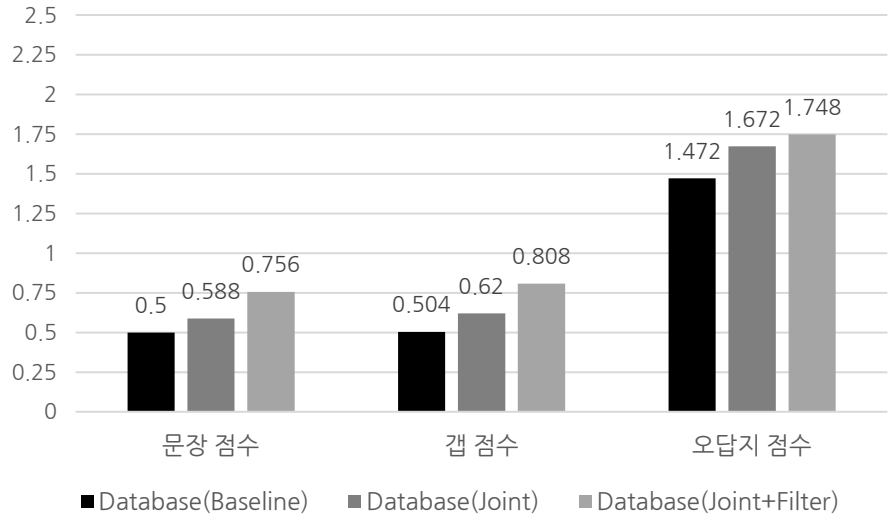

그림 9 데이터베이스 퀴즈 실험 결과

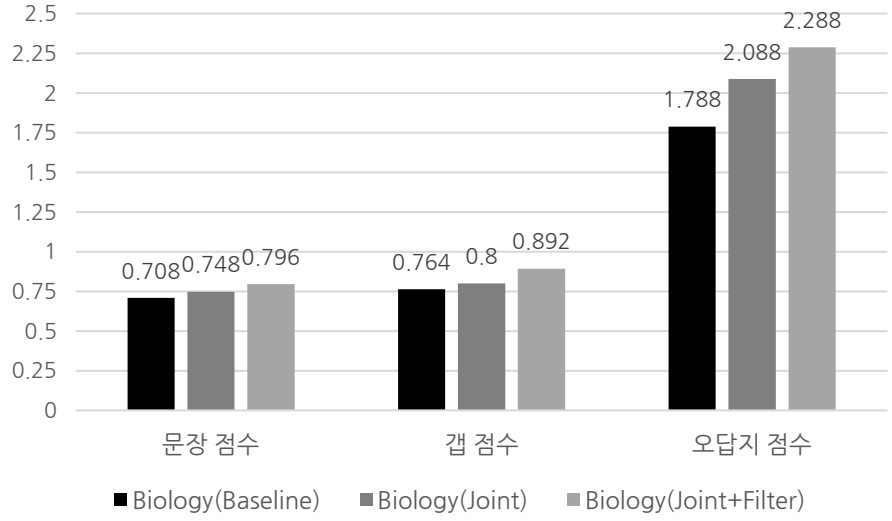

그림 10 생물학 퀴즈 실험 결과

본 논문에서 제안한 유사도 기반 퀴즈 생성 방법으로 생성된 퀴즈(Joint)가 두 책 모두에 대해 기존 연구[7](Baseline)보다 성능이 좋은 것으로 나타났다. 데이터베이스 퀴즈의 경우 문장과 갭에서 모두 0.1정도 높았고, 오답지의 경우 0.2정도 높았다. 생물학 퀴즈의 경우 문장과 갭에서 0.04정도의 소폭의 차이가 있었고 오답지의 경우 0.3정도의 차이가 있었다. 이 결과에 의하면 문장과 갭 선택에 대해서도 유사도 기반 퀴즈 생성 방법이 효과가 있었지만, 오답지 선택에 특히 효과가 있었다는 것을 알 수 있다. 이는 오답지 선택 단계에 있어서 단어의 시맨틱이 결정적인 역할을 하고, 유사도 기반 퀴즈 생성 방법이 이것을 잘 충족시킨 결과로 보인다. 그리고 유사도 기반 퀴즈 생성 방법으로 생성된 퀴즈에 키워드 기반 사후 필터링을 적용한 방법(Joint+Filter)이 필터링을 거치지 않은 방법(Joint)에 비해 성능이 높은 것으로 나타났다. 데이터베이스 퀴즈의 경우 문장과 갭에서는 0.2, 오답지에서는 0.07정도의 효과가 있었으며, 생물학 퀴즈의 경우 문장과 갭에서는 0.1, 오답지에서는 0.2정도의 차이가 있었다. 이 결과는 본 논문에서 제안한 키워드 기반 사후 필터링이 퀴즈 품질 향상에 효과가 있음을 보여준다.

특이한 점은 데이터베이스 책과 생물학 책의 성능 차이가 크게 났다는 것인데, 기존 연구의 방법(Baseline)을 적용했을 때에는 그 차이가 매우 컸으며, 본 논문의 모델(Joint, Joint+Filter)을 적용했을 때에는 차이가 줄긴 했지만 그래도 큰 것을 볼 수 있다. 이는 두 책의 서술 방식의 차이로 보인다. 데이터베이스 책은 특정 개념을 많이 풀어서, 그리고 예시를 많이 들어 설명하는 반면 생물학 책은 개념이나 사실 그 자체를 간단명료

하게 한두 문장으로 나타낸다. 이러한 생물학 책의 서술 방식이 표 3에서 설명한 평가 기준에 적합한 문장과 갭이 많게 하였던 것으로 보인다.

문장, 갭, 오답지에 대해 각각 어떤 문장, 갭, 오답지가 높은 평가를 받고 낮은 평가를 받았는지 살펴보도록 하겠다. 표 4는 평가된 문장 중 일부이다. 첫 번째 문장과 같이 용어(개념)을 설명하거나 두 번째 문장과 같이 사실을 설명한 문장은 좋은 평가를 받았으나, 세 번째 문장과 같이 문장에서 찾을 수 없는 정보(This type)를 가지고 있는 문장이나 네 번째 문장과 같이 책의 구성을 설명하는 등의 잡다한 문장들은 좋은 평가를 받지 못하였다.

표 4 문장 평가의 예

문장	점수 평균
Oracle Enterprise Manager (OEM) is Oracle's main tool for database systems management.	1
A system that processes large transactions can improve response time as well as throughput by performing subtasks of each transaction in parallel.	1
This type of partitioning is useful if the data in the partitioning column have a relatively small set of discrete values.	0
We then cover the tuple relational calculus and the domain relational calculus which are declarative query languages based on mathematical logic.	0.4

표 5은 평가된 갭 중 일부이다. 첫 번째와 두 번째 문장의 갭은 해당 문장에서 설명하고자 하는 핵심 단어이다. 첫 번째 문장은 치료받지 않은 **아테롬성 동맥 경화증**은 종종 심장 마비 또는 뇌졸중을 유발한다는 것이고, 두 번째 문장은 담즙을 생성할 때 간은 적혈구 분해의 부산물인 일부 **색소**를 포함한다는 것으로, 모두 해당 문장에서의 핵심 단어인 것을 알 수 있다. 하지만 세 번째와 네 번째 문장의 갭과 같이 해당 문장의 핵심 단어가 아니거나 흔한 단어들은 좋은 평가를 받지 못했다.

표 5 갭 평가의 예

문장	갭	점수 평균
The result of untreated ____ is often a heart attack or a stroke.	atherosclerosis	1
In producing bile the liver incorporates some ____ that are byproducts of red blood cell disassembly.	pigments	1
Analogous structures that arose independently are also called homoplasies from the Greek ____ to mold in the same way.	meaning	0.2
Chromosomal ____ that delete, disrupt or rearrange many loci at once are usually harmful.	changes	0.2

표 6은 평가된 오답지 중 일부이다. 첫 번째 오답지들의 경우에는 실제로 inner join, left join, right join이 책 상에 존재하기 때문에 비슷한 용도

로 쓰이는 오답지가 잘 선택되어 만점을 받았으며, 두 번째 오답지들의 경우에는 모두 거대 분자 성분들을 이르는 말이어서 2.67의 높은 점수를 받았다. 하지만 세 번째 오답지들의 경우에는 갭과 전혀 상관 없는 오답지들이 선택되어 매우 낮은 점수를 받았다. 실제로 낮은 점수를 받은 오답지들의 경우를 살펴보면 갭이 일반적인 단어인 경우가 많았는데, 이 경우에도 갭이 일반적으로 잘 쓰이는 결합(attraction)이었기 때문에 오답지들이 잘 선택되지 못하였다. 특히 갭의 단어가 일반적으로 쓰이는 의미와 다르게 쓰인 경우에 낮은 평가를 받은 경우가 많았다.

표 6 오답지 평가의 예

문장	오답지	점수 평균
The **outer** join operation works in a manner similar to the join operations we have already studied but preserve those tuples that would be lost in a join by creating tuples in the result containing null values.	1) inner 2) left 3) right	3
Every cell has a vast assortment of **macromolecules** including enzymes and other proteins and the nucleic acids that are essential for self-replication.	1) polymers 2) sugars 3) lipids	2.8
This noncovalent **attraction** between a hydrogen and an electronegative atom is called a hydrogen bond.	1) cue 2) interface 3) vasocongestion	1

제 5장 결론 및 향후 연구

5.1 결론

본 논문에서는 텍스트 임베딩을 이용하는 새로운 객관식 빈칸 채우기 방법을 제안하였다. 기존 빈칸 채우기 퀴즈 생성 연구들이 텍스트의 시맨틱을 경시하고 있다는 점을 해결하기 위하여 본 논문에서는 문서, 문장 그리고 단어 임베딩을 동시에 학습하는 퀴즈 생성에 적합한 임베딩 모델을 제안하였다. 임베딩 모델을 통해 생성된 텍스트 임베딩은 텍스트의 시맨틱을 표현하고 있어 시맨틱을 고려한 퀴즈 생성이 가능해졌다. 그리고 역시 기존 빈칸 채우기 퀴즈 생성 연구들의 한계점인 도메인 종속적인 피쳐와 룰의 사용을 해결하기 위하여 텍스트 임베딩을 기반으로 한 유사도 기반 객관식 빈칸 채우기 퀴즈 생성 방법을 제시하였다. 유사도 기반 퀴즈 생성 방법은 문서-문장, 문장-단어, 단어-단어 간의 벡터 공간상에서의 거리만을 이용하기 때문에 도메인 종속적인 피쳐들이 필요없고, 따라서 다양한 도메인에 범용적으로 쓰일 수 있다는 장점이 있다. 그리고 교과서의 색인 등 미리 정의된 중요한 키워드 리스트가 있는 경우 이를 이용하기 위해 키워드 기반 사후 필터링 방법을 제시하였다. 데이터베이스 책과 생물학 책에 대해 실험한 결과 기존 객관식 빈칸 채우기 퀴즈 연구에 비해 본 논문의 유사도 기반 퀴즈 생성 방법이 10%p~30%p정도 성능이 더 높았으며, 또한 키워드 기반 사후 필터링 방법을 적용하였을 때 10%p~20%p의 성능 향상을 보여 본 논문의 유사도 기반 퀴즈 생성 방법과 키워드 기반 사후 필터링 방법의 성능 향상 효과

를 확인할 수 있었다.

5.2 향후 연구

본 논문에서는 주어진 문서에 대해 문서, 문장 그리고 단어 임베딩을 학습한다. 하지만 주어지는 문서의 크기가 책 한 권 정도의 분량밖에 되지 않아 임베딩 학습의 인풋으로 들어가는 데이터의 크기가 작은 편이다. 이렇게 데이터의 크기가 작으면 임베딩이 제대로 학습되지 않을 수 있는 문제점이 존재한다. 실제로 Mikolov의 Word2Vec 논문[11]에서는 단어의 개수가 6B인 Google News 코퍼스(corpus)[19]와 320M인 LDC 코퍼스[20]를 사용하여 모델을 실험하였다. 하지만 이에 비해 본 논문에서 실험한 데이터베이스 책과 생물학 책의 단어 수는 59만개와 75만개로 [11]에 비해 작은 수준이었다. 본 논문에서는 이러한 어려움에 대응하기 위해 임베딩을 학습할 때 미리 학습된 단어 임베딩들을 초기값으로 사용하였지만, 전공 교과서를 사용했기 때문에 해당 분야에서만 등장하는 용어들이 많아 이러한 단어들은 미리 학습된 단어 임베딩이 없었다. 중요한 갭이 될 만한 단어들 중 많은 경우가 이러하여 임베딩이 충분히 학습되지 못하였다. 향후에는 이러한 문제점을 해결하기 위하여 퀴즈 출제의 대상이 되는 문서에 더해 해당 도메인과 연관된 문서들을 위키피디아 등 웹에서 자동으로 찾아주어 이들 문서와 같이 임베딩을 학습하는 식의 방법을 시도해보고자 한다. 이렇게 한다면 데이터의 크기가 작아 임베딩이 충분이

[19] https://code.google.com/archive/p/word2vec/

[20] https://www.ldc.upenn.edu/

학습되지 못하는 문제점을 해결할 수 있을 것이다.

또한 키워드 기반 사후 필터링 방법을 개선해보고자 한다. 현재는 각 요소, 즉 키워드 리스트, 문서 제목, 그리고 TF-IDF에 대해 점수를 각각 매기고 이를 가중치 평균하여 이를 컷오프(cutoff) 점수로서 사용하였는데, 대신 각 요소의 값을 피쳐로 하여 분류기를 학습한다면 각 요소별 중요도를 감안할 수 있어 좀 더 정밀하게 질 좋은 퀴즈와 질 나쁜 퀴즈를 분류해낼 수 있을 것이다.

참고 문헌

[1] R. Mitkov and L. A. Ha, "Computer-aided generation of multiple-choice tests," in *Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing-Volume 2*, 2003, pp. 17-22.

[2] G. A. Miller, "WordNet: a lexical database for English," *Communications of the ACM,* vol. 38, pp. 39-41, 1995.

[3] M. Heilman and N. A. Smith, "Question generation via overgenerating transformations and ranking," DTIC Document2009.

[4] Y.-T. Huang, Y.-M. Tseng, Y. S. Sun, and M. C. Chen, "TEDQuiz: automatic quiz generation for TED talks video clips to assess listening comprehension," in *Advanced Learning Technologies (ICALT), 2014 IEEE 14th International Conference on*, 2014, pp. 350-354.

[5] G. Erkan and D. R. Radev, "Lexrank: Graph-based lexical centrality as salience in text summarization," *Journal of Artificial Intelligence Research,* vol. 22, pp. 457-479, 2004.

[6] E. Sumita, F. Sugaya, and S. Yamamoto, "Measuring non-native speakers' proficiency of English by using a test with automatically-generated fill-in-the-blank questions," in *Proceedings of the second workshop on Building Educational Applications Using NLP*, 2005, pp. 61-68.

[7] M. Agarwal and P. Mannem, "Automatic gap-fill question generation from text books," in *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, 2011, pp. 56-64.

[8] L. Becker, S. Basu, and L. Vanderwende, "Mind the gap: learning to choose gaps for question generation," in *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2012, pp. 742-751.

[9] A. Nenkova and L. Vanderwende, "The impact of frequency on summarization," *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005,* vol. 101, 2005.

[10] G. Kumar, R. E. Banchs, and L. F. D'Haro, "RevUP: Automatic Gap-Fill Question Generation from Educational Texts," *Silver Sponsor,* pp. 154-161, 2015.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781,* 2013.

[12] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, "Introduction to WordNet: An on-line lexical database," *International journal of lexicography,* vol. 3, pp. 235-244, 1990.

[13] J. R. Firth, "A synopsis of linguistic theory, 1930-1955," *Studies in Linguistic Analysis,* pp. 1-32, 1957.

[14] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research,* vol. 3, pp. 1137-1155, 2003.

[15] Q. V. Le and T. Mikolov, "Distributed Representations of Sentences and Documents," in *ICML*, 2014, pp. 1188-1196.

[16] R. Mihalcea and P. Tarau, "TextRank: Bringing order into texts," 2004.

[17] A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database System Concepts*: McGraw-Hill, 2011.

[18] J. B. Reece, *Campbell Biology*: Benjamin Cummings / Pearson, 2011.

[19] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global Vectors for Word Representation," in *EMNLP*, 2014, pp. 1532-1543.

Abstract

Automatic Generation of Multiple-Choice Gap-Fill Quizzes Using Text Embedding

Junghyuk Park

Department of Computer Science and Engineering

The Graduate School

Seoul National University

Quizzes are widely used in various places and fields. There is a great demand for quizzes not only in quiz shows and quiz competitions but also in ability assessment in various fields and spot of education. However, it takes a lot of time and effort to make these quizzes. Therefore, there is a strong demand for a system that automatically generates such quizzes. In this paper, we propose a method to automatically generate multiple-choice gap-fill quizzes.

Existing automatic multiple-choice gap-fill quiz generation studies have had limitations in 1) neglecting the semantics of text, and 2) using domain-specific features and rules. Therefore, in this paper, we propose a text embedding model that can reflect the semantics of text to solve the problem of 1), and proposed a similarity-based quiz generation method to solve the problem of 2). We also proposed a keyword-based post-filtering method to improve the quality of the final quizzes

when there is a pre-defined keyword list.

Experiments on database book and biology book showed that our similarity-based quiz generation algorithm outperformed the previous study by 10%p~30%p. In addition, when the keyword-based post-filtering method is applied, the performance improvement of 10%p~20%p is confirmed.

Keywords: Automatic Quiz Generation, Word Embedding, Text Embedding, Neural Network Based Embedding Model, Semantic Similarity

Student Number: 2015-22898